TREC 2020 Fair Ranking Track
Participant Instructions

August 19, 2020

For 2020, we will again be adopting an academic search task, where we have a corpus of academic article abstracts and queries submitted to a production academic search engine. The central goal of the Fair Ranking track is to provide *fair exposure* to different groups of authors (a *group fairness* framing). We recognize that there may be multiple group definitions (e.g. based on demographics, stature, topic) and hope for the systems to be robust to these. As such, participants are expected to develop systems to optimize for fairness and relevance for arbitrary group definitions. We will not reveal the exact group definitions until *after* the evaluation runs are submitted.

The track is set up as two tasks, *reranking* and *retrieval*, with a shared evaluation.

Rerank runs will sort a query-dependent list of documents to simultaneously provide fairness and relevance.

Retrieval runs will return 100-item rankings from the corpus in response to a query string.

The track organizers will provide a sequence of queries, each accompanied by a varying-size set of documents. The same queries will be used for both tasks; participants are asked not to use the test queries' rerank sets as a component of their retrieval model training.

1 Protocol

For our fair ranking evaluation, we will be providing participants with a sequence $\mathcal{Q}$ of queries accompanied by unordered sets of documents to rank. The document sets are of varying size. For each request (query q and set of documents $\mathcal{D}_q$), participants should provide a ranked list of the documents from $\mathcal{D}_q$. For the retrieval task, q is the set of all documents in our corpus and participants will be asked to return a fixed set of documents. The final system output is a sequence of rankings. Algorithm 1 presents a pseudocode of the evaluation protocol.

The rankings produced in response to queries in the sequence should balance two goals: (1) be relevant to the consumers and (2) be fair to the producers.

Algorithm 1 Evaluation protocol

$\forall q, \mathcal{D}_q \in \mathcal{Q}, \Pi_q \leftarrow \{\}$
for $q, \mathcal{D}_q \in \mathcal{Q}$ **do**
  $\pi \leftarrow \textsc{System}(q, \mathcal{D}_q)$
  $\Pi_q \leftarrow \Pi_q \cup \{\pi\}$
end for
return $\{\Pi_q\}$

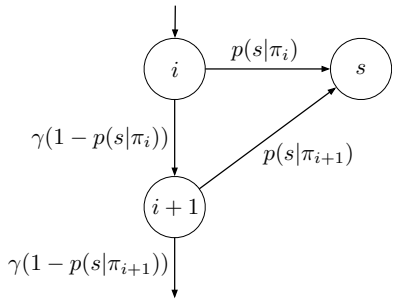

Figure 1: Attention model.

2 Evaluation

Unlike previous TREC tracks, you will receive multiple copies or *impressions* of the same query text–although the query id will be different–and you may submit different rankings for each query impression. At evaluation time, we will measure *expected exposure* of groups over rankings produced for each given query [2].

Given a sequence of queries $\mathcal{Q}$ and associated system rankings, we will be evaluating systems according to fair exposure of authors and relevance of documents.

2.1 Measuring Author Exposure for a Single Query

In order to measure exposure, we adopt the browsing model underlying the Expected Reciprocal Rank metric [1]. Given a static ranking π in response to a query impression, the exposure of author a is,

$$e_a^\pi = \sum_{i=1}^{n} \left[\gamma^{i-1} \prod_{j=1}^{i-1} (1 - p(s|\pi_j)) \right] I(\pi_i \in \mathcal{D}_a)$$

n	number of documents in ranking π
$\mathcal{D}_a$	documents including a as an author
π_i	document at position i
γ	continuation probability (fixed to 0 for the final position in the ranking)
$p(s\|d)$	probability of stopping given user examined d

We present a graphical depiction of this model in Figure 1. We will use a discounting factor $\gamma = 0.5$, and will assume $p(s|d) = f(r_d)$, where r_d is the relevance of the document d and f is a monotonic transform of that relevance into a probability of being satisfied.

A unique query will have a number of impressions. In order to compute the *expected exposure* for a, we consider the set of all rankings presented by the system for that query Π_q,

$$e_a = \sum_{\pi \in \Pi_q} e_a^\pi \tag{1}$$

The *target expected exposure* for a query is derived from Equation 1 assuming a policy that randomizes amongst all permutations whose relevance monotonically degrades with rank [2].

2.2 Measuring Group Exposure for a Single Query

Assume that each author is assigned to exactly one of $|\mathcal{G}|$ groups. Let $\mathcal{A}_g$ be the set of all authors in group g. The group expected exposure is defined as,

$$\mathcal{E}_g = \sum_{a \in \mathcal{A}_g} e_a \tag{2}$$

We define the target group expected exposure $\mathcal{E}_g^*$ as Equation 2 using the individual target expected exposure (Section 2.1).

2.3 Expected Exposure Metric

We will evaluate systems using the per-query difference in system group expected exposure and target group expected exposure,

$$\left(\sum_{g \in \mathcal{G}} (\mathcal{E}_g - \mathcal{E}_g^*)^2 \right)^{\frac{1}{2}} \tag{3}$$

Per-query metrics will be averaged to compute the summary metric for the run.

3 Data

3.1 Input

There are three main inputs available to you: the *corpus* of articles to search, the *example group definition* file to help you develop and test your solution, and the *queries*.

3.1.1 Paper and Author Data

The paper and author metadata CSV files we will be providing with the query data provide summary information for papers and their authors in the rerank set. There are three files:

paper_metadata.csv contains basic paper information: ID, title, year, venue, and the number of citations.

author_metadata.csv contains author information: ID, name, citation count, paper count, and H-index.

authors_for_papers.csv contains the author list for each paper: paper ID, author ID, and position.

These files do *not* contain abstracts. To create a usable text index, you will need the corpus in the next section.

3.1.2 Corpus

The full corpus for this project is the Semantic Scholar (S2) Open Corpus from the Allen Institute for Artificial Intelligence. It can be downloaded from `http://api.semanticscholar.org/corpus/`, and consists of 47 1GB data files. Each file is compressed JSON, where each line is a JSON object describing one paper. The following data are available for most papers:

- S2 Paper ID
- DOI
- Title

- Abstract
- Authors (resolved to author IDs)
- Inbound and outbound citations (resolved to S2 paper IDs)

3.1.3 Example Group Definition Data

To help you get started, we have provided the file `fair-TREC-sample-author-groups.csv` containing group ids for authors in the S2 corpus. This group definition will not be our final group definition (we will evaluate using multiple various group definitions), but you can use it to start working on the task.

This CSV file contains two columns:

1. The `author` column has the S2 ID of the author.
2. The `gid` column has the author's group identifier.

3.1.4 Queries

The training queries are in `fair-TREC-training-sample.json`; this is a JSON file where each line is a JSON object (a dictionary) describing one query:

1. Query ID ('qid')
2. Query string ('query')
3. Query frequency ('frequency')
4. A list of documents ('documents') with relevance information; this is a list of dictionaries with two keys: 'doc_id' and 'relevance'

Runs will be submitted over *query sequences*: ordered sequences of queries that may contain duplicates. We will provide the set of query sequences for official runs 1 month prior to the deadline; for training and development, use the provided script (`query-sequence-generator.py`) to generate query sequences from the training queries distributions.

A query sequence is a CSV file with the following fields:

1. <sequence id>.<query number in sequence>
2. <query id> (to look up in query file)

Please remember to treat each sequence in the file separately. You should *not* treat this as a single sequence to operate on.

The evaluation query sequences will be accompanied by a query file that has no relevance information and has all query frequencies set to 0, to enable you to only look up the queries and document sets to re-rank.

3.2 Output

For each query sequence, your submitted ranking will be a JSON file where each line is a JSON object (a dictionary) containing your ranking results:

- <sequence id>.<query number in sequence> ('q_num')
- <query id> (to look up in query file) ('qid')
- An ordered list of document IDs (of the documents to be re-ranked for the query) ('ranking')

References

[1] O. Chapelle, D. Metzler, Y. Zhang, and P. Grinspan. Expected reciprocal rank for graded relevance. In *Proceedings of the 18th ACM conference on Information and knowledge management*, CIKM '09, pages 621–630, New York, NY, USA, 2009. ACM.

[2] F. Diaz, B. Mitra, M. D. Ekstrand, A. J. Biega, and B. Carterette. Evaluating stochastic rankings with expected exposure, 2020.